Toolkits for Latent Dirichlet Allocation

UIUC | CS 410: Text Information Systems | Fall 2020
Mihir Yerande (yerande2)

In this article, I give an overview of the model known as *Latent Dirichlet Allocation* (or *LDA*). Though LDA is a general approach that can be applied to collections of discrete data in any area of interest, I specifically focus on LDA in the context of text modeling and analysis. In that space, LDA is used as a method for discovering a set of *topics* among a given collection of text data. In addition, I provide a selection of toolkits which implement LDA, and I compare these at a glance.

[1] LDA Overview

Latent Dirichlet Allocation is a *generative probabilistic model* for text data. In simpler terms, LDA is one conceptualization (a *model*) of text data whereby one imagines that the text was *generated* according to a specific set of rules involving *probability*. LDA is specifically of interest in text analysis because it involves a set of *topics*, where the term *topic* in this context is intended to match the colloquial, human-understandable idea of a topic, such as *cooking*, *travel*, *sports*, etc. This is an enticing model for text analysis because insight into such topics would be extremely useful for a human user. For instance, it would be easier to search through a collection of documents by browsing a ranking of the most relevant documents within a topic of interest.

To see the inner workings of the model, we walk through the process of generating text data with our model. Note that text data is viewed here as a “bag of words”, meaning we only think of a single document as a collection of all of its words, rather than a sequence. First, we have a set of *k* topics. A topic is defined here as a probabilistic weighting of words. For example, we can imagine a *sports*-related topic as one where there is a relatively higher weighting of terms such as *team*, *game*, or *tournament*, in comparison to English text as a whole. Next, when we go to generate a new document, we determine a *topic coverage* for our new document, which is a relative weighting of the topics to be used for generation of this document, such as 70% *cooking* and 30% *travel*. Finally, this weighted mixture of topics is used to probabilistically generate the sequence of words comprising our new document. In this sense, the model has gone through three distinct levels of probabilistic generation to produce a document.

However, this is merely a model of the text data. Simply imagining that the text was generated according to this model does not yet illuminate the parameters of the model, as we initially only observe the text output of it. Thus, the purpose of defining this model is actually to use the observed text data to estimate the remaining unknown configuration of the model, i.e. the *topics* and the *topic coverage*. There are different statistical inference approaches for estimating the model parameters, but in the end, the result is a set of *k* discovered topics shared among the text data, as well as the *topic coverage* of each document in the collection.

[2] Toolkits

This section will discuss a selection of implementations of LDA in modern toolkits.

[2.1] Stanford Topic Modeling Toolbox (TMT)

The Stanford Topic Modeling Toolbox (TMT) is a collection of topic modeling tools produced in 2009 by members of the Stanford Natural Language Processing Group. Since it was written a decade ago and is no longer actively supported, it is not advisable to attempt to integrate this into a larger project simply due to the use of old versions of Scala and its libraries. However, it is still possible to use the software as a standalone tool for modeling given data. On the whole, the toolbox mostly seems useful for the sake of running an ad hoc analysis of topics, rather than for use in a larger programmatic process that might use the model's output.

The toolbox takes text data stored in spreadsheets as input and then produces output in the form of *txt* and *csv* files. The software will preprocess the given text data according to the user's choices. For example, the tokenizer can be configured to prune away non-alphanumeric strings, filter for a minimum length, prune out the top 30 most frequent words, prune out the top 5 least frequent words, etc. These would most likely need to be determined by the user, based on the characteristics of the provided text data. Also, the LDA parameters, such as number of desired topics, can be configured by the user.

The toolbox of course also allows for choice of LDA configuration, with the main choice being *k*, the number of topics to discover. In addition, there is a choice between the numerical method of estimation: Collapsed Gibbs sampling (*GibbsLDA*) vs. Collapsed Variational Bayes approximation (*CVB0LDA*, the default). These are both implemented to take advantage of multi-threading and multi-core machines. However, CVB0LDA requires fewer iterations as it has a faster rate of convergence, but *GibbsLDA* requires less memory during the course of training. Finally, there are also two additional versions of LDA available: Labeled LDA and Partially Labeled LDA (PLDA).

[2.2.1] MALLET

MALLET (the MAchine Learning for LanguagE Toolkit) is a Java package offering various natural language processing and text-related machine learning methods and models. The toolkit can be used for *document classification*, *sequence tagging*, and *topic modeling*, among other applications. The toolkit includes some built-in functionality for tokenization and text processing as well as numerical optimization, which are helpful for topic modeling. The package was primarily written by Prof. Andrew McCallum, with the help of other graduate students, at the University of Massachusetts Amherst. MALLET is generally considered a well-documented and respected implementation of LDA, even though newer implementations have arisen.

In terms of topic modeling, MALLET offers Latent Dirichlet Allocation, Pachinko Allocation, and Hierarchical LDA. The estimation is performed using a multi-threaded implementation of Gibbs sampling which has generally been regarded as fast. MALLET is otherwise similar to other toolkits in that the user would configure the importing and processing of text data, as well as simple parameters such as the desired number of topics. However, MALLET is notable in that it allows for periodic optimization of the hyperparameters, α and β.

[2.2.2] GUI Topic Modelling Tool (GTMT)

The GUI Topic Modeling Tool (GTMT) is a graphical user interface for performing LDA, and its main usage is in running simple LDA on a set of documents for the simple analysis. GTMT is built using the functionality of MALLET, and so is simply a more user-friendly extension of it for less technical users. The output of running the estimation can be viewed as nicely formatted HTML files, which provides for simple browsing and visualization of the topics.

[2.3] Vowpal Wabbit

In 2010, Matt Hoffman published a paper for an online method of LDA which greatly sped up the calculation and allowed for it to not all occur simultaneously in-memory. This was originally written in Python, but then re-implemented in C++ and integrated as part of Vowpal Wabbit (*VW*). The underlying algorithm for estimation, as in Matt's paper, is the Variational Bayes method, which has previously been mentioned above. VW is an open-source library containing implementations of many online machine learning algorithms written in C++. The library originated in *Yahoo! Research* but later came under the ownership of *Microsoft Research*. Though this implementation is considered optimal for a lot of reasons, it is most likely not popular simply due to it being implemented in C++, as opposed to Python. However, the Python library Gensim provides a wrapper around this implementation, which is preferable for most developers.

[2.4] Gensim

Gensim seems to be the most popular library for LDA. It is a Python library which was initially released in 2009 by Radim Řehůřek as a hobby project. With the recent growth of Python in the space of machine learning and data science, the library has become a cornerstone of text analysis and is reputed to be very fast and easy to use. In addition, it wraps the LDA functionality provided in both MALLET and Vowpal Wabbit, which serves to bring those time-hardened implementations into a more temporally relevant programming language. This is one of the major benefits of Gensim. Aside from those, Gensim also has its own implementation of LDA whose estimation algorithm is based on the original Python code written as part of Matt Hoffman's 2010 paper. The algorithm is online, runs in constant memory (w.r.t to number of documents provided), and makes use of distribution where possible.

[3] Conclusion

Latent Dirichlet Allocation is an extremely useful model for analyzing text data through the lens of topics. There are different algorithms for estimating the underlying set of topics, and there are benefits and drawbacks observed in various implementations of those algorithms. Though I have highlighted some popular implementations of LDA, it is possible to find many other ones produced by independent developers. There will likely be other improvements or wholly separate implementations of LDA and its variations emerging over time. Among the toolkits presented here, *Gensim* appears to be the best choice, as it is implemented in Python, which is increasingly becoming more popular in the realm of data science and machine learning. Also, it anyway provides wrappers around the previously popular implementations seen in MALLET and VW. Due to the open-source nature of code nowadays, and the speed at which new frameworks and implementations emerge, one would expect to see progressively more improvements on the algorithms and perhaps even entirely new models which might supplant LDA entirely. LDA is just one part of the very rich landscapes within data science.

References

Blei, David M., et al. “Latent Dirichlet Allocation.” *Journal of Machine Learning Research*, vol. 3, no. 1, Jan. 2003, pp. 993–1022. *ACM Digital Library*, 10.1162/jmlr.2003.3.4-5.993.

Hoffman, Matthew D., et al. “Online Learning for Latent Dirichlet Allocation.” *Advances in Neural Information Processing Systems*, vol. 23, 2010, pp. 856–864.

McCallum, Andrew Kachites. “MALLET Homepage.” *MAchine Learning for LanguagE Toolkit*, UMass Amherst, 2018, mallet.cs.umass.edu/. Accessed 15 Nov. 2020.

Ramage, Daniel, and Evan Rosen. “Stanford Topic Modeling Toolbox.” *The Stanford Natural Language Processing Group*, The Stanford Natural Language Processing Group, Sept. 2009, nlp.stanford.edu/software/tmt/tmt-0.4/. Accessed 15 Nov. 2020.

Řehůřek, Radim. “Models.Ldamodel - Latent Dirichlet Allocation.” *Gensim: Topic Modelling for Humans*, Radim Řehůřek, 2009, radimrehurek.com/gensim/models/ldamodel.html. Accessed 15 Nov. 2020.

“Topic Modeling Tool.” *Google Code*, 22 Sept. 2011, code.google.com/archive/p/topic-modeling-tool/. Accessed 15 Nov. 2020.

“Vowpal Wabbit.” *Vowpal Wabbit*, Microsoft Research, vowpalwabbit.org/. Accessed 15 Nov. 2020.